An Open Web Platform for Rule-Based Speech-to-Sign Translation

Rayner, Manny ; Bouillon, Pierrette ; Ebling, Sarah ; Gerlach, Johanna ; Strasly, Irene ; Tsourakis, Nikos

An Open Web Platform for Rule-Based Speech-to-Sign Translation

Manny Rayner, Pierrette Bouillon, Johanna Gerlach, Irene Strasly, Nikos Tsourakis
University of Geneva, FTI/TIM, Switzerland
{Emmanuel.Rayner,Pierrette.Bouillon,Johanna.Gerlach}@unige.ch
{Irene.Strasly,Nikolaos.Tsourakis}@unige.ch

Sarah Ebling
University of Zurich, Institute of Computational Linguistics, Switzerland
ebling@ifi.uzh.ch

Abstract

We present an open web platform for developing, compiling, and running rule-based speech to sign language translation applications. Speech recognition is performed using the Nuance Recognizer 10.2 toolkit, and signed output, including both manual and non-manual components, is rendered using the JASigning avatar system. The platform is designed to make the component technologies readily accessible to sign language experts who are not necessarily computer scientists. Translation grammars are written in a version of Synchronous Context-Free Grammar adapted to the peculiarities of sign language. All processing is carried out on a remote server, with content uploaded and accessed through a web interface. Initial experiences show that simple translation grammars can be implemented on a time-scale of a few hours to a few days and produce signed output readily comprehensible to Deaf informants. Overall, the platform drastically lowers the barrier to entry for researchers interested in building applications that generate high-quality signed language.

1 Introduction

While a considerable amount of linguistic research has been carried out on sign languages to date, work in automatic sign language processing is still in its infancy. Automatic sign language processing comprises applications such as sign language recognition, sign language synthesis, and sign language translation (Sáfár and Glauert, 2012). For all of these applications, drawing on the expertise of native signers, sign language linguists and sign language interpreters is crucial. These different types of sign language experts may exhibit varying degrees of computer literacy. In the past, their contribution to the development of systems that automatically translate into sign language has been restricted mostly to the provision of transcribed and/or annotated sign language data.

In this paper, we report on the development and evaluation of a platform that allows sign language experts with modest computational skills to play a more active role in sign language machine translation. The platform enables these users to independently develop and run applications translating speech into synthesized sign language through a web interface. Synthesized sign language is presented by means of a signing avatar. To the best of our knowledge, our platform is the first to facilitate low-threshold speech-to-sign translation, opening up various possible use cases, e.g. that of communicating with a Deaf customer in a public service setting like a hospital, train station or bank.[1] By pursuing a rule-based translation approach, the platform also offers new possibilities for empirical investigation of sign language linguistics: the linguist can concretely implement a fragment of a hypothesized sign language grammar, sign a range of generated utterances through the avatar, and obtain judgements from Deaf informants.

The remainder of this paper is structured as follows. Section 2 presents background and related work. Section 3 describes the architecture of the speech-to-sign platform. Section 4 reports on a preliminary evaluation of the usability of the platform and of translations produced by the platform. Section 5 offers a conclusion and an outlook on future research questions.

[1] We follow the widely recognized convention of using the upper-cased word *Deaf* to describe members of the linguistic community of sign language users and, in contrast, the lower-cased word *deaf* to describe the audiological state of a hearing loss (Morgan and Woll, 2002).

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 162–168,
Berlin, Germany, August 7-12, 2016.

2 Background and related work

There has been surprisingly little work to date on speech to sign language translation. The best-performing system reported in the literature still appears to be TESSA (Cox et al., 2002), which translated English speech into British Sign Language (BSL) in a tightly constrained post office counter service domain, using coverage captured in 370 English phrasal patterns with associated BSL translations. The system was evaluated in a realistic setting in a British post office, with three post office clerks on the hearing side of the dialogues and six Deaf subjects playing the role of customers, and performed creditably. Another substantial project is the one described by San-Segundo et al. (2008), which translated Spanish speech into Spanish Sign Language; this, however, does not appear to have reached the stage of being able to achieve reasonable coverage even of a small domain, and the evaluation described in the paper is restricted to comprehensibility of signs from the manual alphabet.[2]

It is reasonable to ask why so little attention has been devoted to what many people would agree is an important and interesting problem, especially given the early success of TESSA. Our own experiences, and those of other researchers we have talked to, suggest that the critical problem is the high barrier to entry: in order to build a speech-to-sign system, it is necessary to be able to combine components for speech recognition, translation and sign language animation. The first two technologies are now well-understood, and good platforms are readily available. Sign language animation is still, however, a niche subject, and the practical problems involved in obtaining usable sign language animation components are non-trivial. The fact that San-Segundo et al. (2008) chose to develop their own animation component speaks eloquently about the difficulties involved.

There are three approaches to sign language animation: hand-crafted animation, motion capturing and synthesis from form notation (Glauert, 2013). Hand-crafted animation consists of manually modeling and posing an avatar character. This procedure typically yields high-quality results but is very labor-intensive. A signing avatar may also be animated based on information obtained from motion capturing, which involves recording a human's signing. Although sign language animations obtained through motion capturing also tend to be of good quality, the major drawback of this approach is the long calibration time and extensive postprocessing required.

Synthesis from form notation permits construction of a fully-fledged animation system that allows synthesis of any signed form that can be described through the associated notation. Avatar signing synthesized from form notation is the most flexible in that it is able to render dynamic content, e.g. display the sign language output of a machine translation system, present the contents of a sign language wiki or an e-learning application, visualize lexicon entries or present public transportation information (Efthimiou et al., 2012; Kipp et al., 2011). At the same time, this approach to sign language animation typically results in the lowest quality: controlling the appearance of all possible sign forms that may be produced from a given notation is virtually impossible.

The most comprehensive existing sign language animation system based on synthesis from form notation is undoubtedly JASigning (Elliott et al., 2008; Jennings et al., 2010), a distant descendant of the avatar system used in TESSA which was further developed over the course of the eSIGN and DictaSign European Framework projects. JASigning performs synthesis from SiGML (Elliott et al., 2000), an XML-based representation of the physical form of signs based on the well-understood Hamburg Notation System for Sign Languages (HamNoSys) (Prillwitz et al., 1989). HamNoSys can be converted into SiGML in a straightforward fashion. Unfortunately, despite its many good and indeed unique properties, JASigning is a piece of research software that in practice has posed an insurmountable challenge to most linguists without a computer science background.

The basic purpose of the Lite Speech2Sign project can now be summarised in a sentence: we wished to package JASigning together with a state-of-the-art commercial speech recognition platform and a basic machine translation framework in a way that makes the combination easily usable by sign language linguists who are not software engineers. In the rest of the paper, we describe the result.

[2] Sign languages make use of a communication form known as the *manual alphabet* (or, *finger alphabet*), in which the letters of a spoken language word are fingerspelled, i.e., dedicated signs are used for each letter of the word.

3 The Lite Speech2Sign platform

The fact that the Lite Speech2Sign platform is intended primarily for use by sign language experts who may only have modest skills in computer science has dictated several key design decisions. In particular, 1) the formalism used is simple and minimal and 2) no software need be installed on the local machine: all processing (compilation, deployment, testing) is performed on a remote server accessed through the web interface.

3.1 Runtime functionality and formalism

At runtime, the basic processing flow is speech → source language text → "sign table" → SiGML → signed animation. Input speech, source language text and signed animation have their obvious meanings, and we have already introduced SiGML in the preceding section. At the input end of the pipeline, speech recognition is carried out using the Nuance Recognizer 10.2 platform, equipped with domain-specific language models compiled from the grammar. At the output end, SiGML is converted into signed animation form using the JASigning avatar system.

The "sign table", the level which joins all these pieces together, is an intermediate representation modelled on the diagrams typically used in theoretical sign language linguistics to represent signed utterances. A sign table is, concretely, a matrix whose rows represent the different parallel channels of signed language output (manual activities, gaze, head movements, mouth movements, etc). The only obligatory row is the one for manual activities, which consists of a sequence of "glosses", each gloss referring to one manual activity. There is one column for each gloss/manual activity in the signed utterance.

The usefulness of this representation is dependent on the appropriateness of the assumption that sign language is timed so that each non-manual activity can be assumed synchronous with some manual activity. This has been shown to be true for non-manual activities that serve linguistic functions. Non-manual activities that serve purely affective purposes, e.g., expressing anger or disgust, are known to start slightly earlier than the surrounding manual activities (Reilly and Anderson, 2002; Wilbur, 2000). A restriction imposed by the low-level SiGML representation is that non-manual activities cannot be extended across several manual activities in a straightforward way; however, workarounds have been introduced for this (Ebling and Glauert, 2015). Experience with SiGML has shown that it is capable of supporting signed animation of satisfactory quality (Smith and Nolan, 2015).

```
include lsf_ch.csv
include visicast.txt

Domain
Name toy1
Client speech2sign_client
SourceLanguage french
TargetLanguages gloss head gaze \
    eyebrows aperture mouthing
EndDomain

Utterance
Source   je m'appelle $$name
Gloss    MOI     S_APPELER $$name
Head     Nod     Neutral   Neutral
Gaze     Neutral Neutral   Neutral
Eyebrows Up      Up        Up
Aperture Wide    Wide      Wide
Mouthing mwe     appel     $$name
EndUtterance

TrPhrase $$name
Source    claude
Gloss     C L A U  D E
Mouthing  C L a u: d e
EndTrPhrase

TrPhrase $$name
Source    marie
Gloss     M   A R I E
Mouthing  L23 a R i e
EndTrPhrase
```

Figure 1: Toy speech2sign application definition.

The core translation formalism is a version of Synchronous Context Free Grammar (SCFG; (Aho and Ullman, 1969; Chiang, 2005)) adapted to the peculiarities of sign language translation. A complete toy application definition is shown in Figure 1. The top-level `Utterance` rule translates French expressions of the form *Je m'appelle* ⟨*NAME*⟩ ("I am called ⟨NAME⟩") to Swiss French Sign Language (LSF-CH) expressions of a form

that can be glossed as *MOI S_APPELER* ⟨*NAME*⟩ together with accompanying non-manual components; for example, the manual activity *MOI* (signed by pointing at one's chest) is here performed together with a head nod, raised eyebrows, widened eyes, and a series of mouth movements approximating the shapes used to say "mwe". The two `TrPhrase` rules translate the names "Claude" and "Marie" into fingerspelled forms with accompanying mouthings.

The mapping between the sign table and SiGML levels is specified using three other types of declarations, defined in the resource lexica listed in the initial `include` lines. 1) Glosses are associated with strings of HamNoSys symbols; in this case, the resource lexicon used is `lsf_ch.csv`, a CSV spreadsheet whose columns are glosses and HNS strings for LSF-CH signs. 2) Symbols in the non-manual rows (`Head`, `Gaze`, etc) are mapped into the set of SiGML tags supported by the avatar, according to the declarations in the sign-language-independent resource file `visicast.txt`. 3) The `Mouthing` line is treated specially. Two types of mouthings are supported: "mouth pictures", approximate mouthings of phonemes, are written as SAMPA (Wells, 1997) strings (e.g. `mwe` is a SAMPA string). It is also possible to use the repertoire of "mouth gestures" (mouth movements not related to spoken language words, produced with teeth, jaw, lips, cheeks, or tongue) supported by the avatar, again using definitions taken from the `visicast.txt` resource file. For example, `L23` denotes pursed lips (Hanke, 2001).

The `Domain` unit at the top defines the name of the translation app, the source language[3] and sign language channels, and the type of web client used to display it.

3.2 Compile- and deploy-time functionality

The compilation process takes application descriptions like the one above as input and transforms them first into SCFG grammars, then into GrXML grammars[4], and finally into runnable Nuance recognition grammars. The compiler also produces tables of metadata listing associations between symbols and HamNoSys, SAMPA, and SiGML constants.

Two main challenges needed to be addressed when designing the compile-time functionality. The first was to make the process of developing, uploading, compiling, and deploying web-based speech applications simple to invoke, so that these operations could be performed without detailed understanding of the underlying technology. The second was to support development on a shared server; here, it is critical to ensure that a developer who uploads bad content is not able to break the system for other users.

At an abstract level, the architecture is as follows. Content is divided into separate "namespaces", with each developer controlling one or more namespaces; a namespace in turn contains one or more translation apps. At the source level, each namespace is a self-contained directory, and each app a self-contained subdirectory.

From the developer's point of view, the whole upload/compile/deploy cycle reduces to a simple progression across a dashboard with four tabs labeled "Select", "Compile", "Test", and "Release". The developer starts the upload/compile/deploy cycle by uploading one or more namespace directories over an FTP client and choosing one of them from the "Select" tab.

The platform contains three separate servers, respectively called *compilation*, *staging*, and *deployment*. After selecting the app on the first tab, the developer moves to the second one and presses the "Compile" button to invoke the compilation server. Successful compilation results in a Nuance grammar recognition module and a set of namespace-specific table entries; a separate Nuance recognition grammar is created for each namespace. As part of the compilation process, a set of files is also created which list undefined constants. These can be downloaded over the FTP connection and are structured so as to make it easy for the developer to fill in missing entries and add the new content to the resource files.

When the app has compiled, the developer proceeds to the third, "Staging" tab, and presses the "Test" button. This initiates a process which copies the compiled recognition grammar, table entries and metadata to appropriate places on the staging server and registers the grammar as available for use by the recognition engine, after which the developer can interactively test the application

[3] Any recognition language supported by Nuance Recognizer 10.2 can potentially be used as a source language; the current version of the platform is loaded with language packs for English, French, German, Italian, Japanese and Slovenian.

[4] GrXML is an open standard for writing speech recognition grammars.

through the web interface. It is important that only copying actions are performed by the "Staging" server; experience shows that recompiling applications can often lead to problems if the compiler changes after an application is uploaded.

When the developer is satisfied with the application, they move to the fourth tab and press the "Release" button. This carries out a second set of copying operations which transfer the application to the deployment server.

4 Initial experiences with the platform

The Lite Speech2Sign platform is undergoing initial testing; during this process, we have constructed half a dozen toy apps for the translation directions French → LSF-CH and German → Swiss German Sign Language, and one moderately substantial app for French → LSF-CH. Grammars written so far all have a flat structure.

Our central claims regarding the platform are that it greatly simplifies the process of building a speech-to-sign application and allows rapid construction of apps which produce signed language of adequate quality. To give some substance to these statements, we tracked the construction of a small French → LSF-CH medical questionnaire app and performed a short evaluation. The app was built by a sign language expert whose main qualifications are in sign language interpretation. The expert began by discussing the corpus with Deaf native signers, to obtain video-recorded material on which to base development. They then implemented rules and HNS entries, uploaded, debugged, and deployed the content, and used the deployed system to perform the evaluation.

Rule-writing typically required on the order of ten to fifteen minutes per rule, using a method of repeatedly playing the recorded video and entering first the gloss line and then the accompanying non-manual lines. Uploading, debugging, and deployment of the app was completely straightforward and took approximately one hour. The most time-consuming part of the process was implementing HNS entries for signs missing from the current LSF-CH HNS lexicon. The time required per entry varied a great deal depending on the sign's complexity, but was typically on the order of half an hour to two hours. This part of the task will of course become less important as the HNS lexicon resource becomes more complete.

The evaluation was carried out with five Deaf subjects and based on recommendations for sign language animation evaluation studies by Kacorri et al. (2015). Each subject was first given a short demographic questionnaire. Subjects were then asked to watch seven outputs from the app and echo them back, either in signed or mouthed form, to check the comprensibility of the app's signed output. They then answered a second short questionnaire which asked for their overall impressions. The result was encouraging: although none of the subjects felt the signing was truly fluent and human-like (a frequent comment was "artificial"), they all considered it grammatically correct and perfectly comprehensible.

5 Conclusions and further directions

Although the Lite Speech2Sign platform is designed to appear very simple and most of its runtime processing is carried out by the third-party JASigning and Nuance components, it represents a non-trivial engineering effort. The value it adds is that it allows sign language linguists who may have only modest computational skills to build translation applications that produce synthesized signed language, using a tool whose basic functioning can be mastered in two or three weeks. By including speech recognition, these applications can potentially be useful in real situations.

In a research context, the platform opens up new possibilities for investigation of the grammar of signed languages. If the linguist wishes to investigate the productivity of a hypothesized syntactic rule, they can quickly implement a grammar fragment and produce a set of related signed utterances, all signed uniformly using the avatar. Our initial experiences, as described in Section 4, suggest that rendering quality is sufficient to obtain useful signer judgements.

Full documentation for Lite Speech2Sign is available (Rayner, 2016). The platform is currently in alpha testing; we plan to open it up for general use during Q3 2016. People interested in obtaining an account may do so by mailing one of the authors of this paper.

Acknowledgements

We would like to thank John Glauert of the School of Computing Sciences, UEA, for his invaluable help with JASigning, and Nuance Inc for generously making their software available to us for research purposes.

References

Alfred V. Aho and Jeffrey D. Ullman. Properties of syntax directed translations. *Journal of Computer and System Sciences*, 3(3):319–334, 1969.

David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics, 2005.

Stephen Cox, Michael Lincoln, Judy Tryggvason, Melanie Nakisa, Mark Wells, Marcus Tutt, and Sanja Abbott. Tessa, a system to aid communication with deaf people. In *Proceedings of the fifth international ACM conference on Assistive technologies*, pages 205–212. ACM, 2002.

Sarah Ebling and John Glauert. Building a Swiss German Sign Language avatar with JASigning and evaluating it among the Deaf community. *Universal Access in the Information Society*, pages 1–11, 2015. Retrieved from `http://dx.doi.org/10.1007/s10209-015-0408-1` (last accessed November 20, 2015).

Eleni Efthimiou, Stavroula-Evita Fotinea, Thomas Hanke, John Glauert, Richard Bowden, Annelies Braffort, Christophe Collet, Petros Maragos, and François Lefebvre-Albaret. The Dicta-Sign Wiki: Enabling web communication for the Deaf. In *Proceedings of the 13th International Conference on Computers Helping People with Special Needs (ICCHP)*, pages 205–212, Linz, Austria, 2012.

Ralph Elliott, John RW Glauert, JR Kennaway, and Ian Marshall. The development of language processing support for the ViSiCAST project. In *Proceedings of the fourth international ACM conference on Assistive technologies*, pages 101–108. ACM, 2000.

Ralph Elliott, John RW Glauert, JR Kennaway, Ian Marshall, and Eva Safar. Linguistic modelling and language-processing technologies for avatar-based sign language presentation. *Universal Access in the Information Society*, 6(4): 375–391, 2008.

John Glauert. Animating sign language for Deaf people. Lecture held at the University of Zurich, October 9, 2013 (unpublished), 2013.

Thomas Hanke. ViSiCAST Deliverable D5-1: Interface definitions. Technical report, ViSiCAST project, 2001. Retrieved from `http://www.visicast.cmp.uea.ac.uk/Papers/ViSiCAST_D5-1v017rev2.pdf` (last accessed November 20, 2015).

Vince Jennings, Ralph Elliott, Richard Kennaway, and John Glauert. Requirements for a signing avatar. In *Proceedings of the 4th LREC Workshop on the Representation and Processing of Sign Languages*, pages 133–136, La Valetta, Malta, 2010.

Hernisa Kacorri, Matt Huenerfauth, Sarah Ebling, Kasmira Patel, and Mackenzie Willard. Demographic and experiential factors influencing acceptance of sign language animation by Deaf users. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, pages 147–154. ACM, 2015.

Michael Kipp, Alexis Heloir, and Quan Nguyen. Sign language avatars: Animation and comprehensibility. In *Proceedings of the 11th International Conference on Intelligent Virtual Agents (IVA)*, pages 113–126, Reykjavík, Iceland, 2011.

Gary Morgan and Bencie Woll. The development of complex sentences in British Sign Language. In Gary Morgan and Bencie Woll, editors, *Directions in Sign Language Acquisition: Trends in Language Acquisition Research*, pages 255–276. John Benjamins, Amsterdam, Netherlands, 2002.

Siegmund Prillwitz, Regina Leven, Heiko Zienert, Thomas Hanke, and Jan Henning. *HamNoSys: Version 2.0: An Introductory Guide*. Signum, Hamburg, Germany, 1989.

Manny Rayner. *Using the Regulus Lite Speech2Sign Platform*. `http://www.issco.unige.ch/en/research/projects/Speech2SignDoc/build/html/index.html`, 2016. Online documentation.

J. Reilly and D. Anderson. FACES: The acquisition of non-manual morphology in ASL. In G. Morgan and B. Woll, editors, *Directions in Sign Language Acquisition*, pages 159–181. John Benjamins, Amsterdam, Netherlands, 2002.

Eva Sáfár and John Glauert. Computer modelling. In Roland Pfau, Markus Steinbach, and Bencie

Woll, editors, *Sign Language: An International Handbook*, pages 1075–1101. De Gruyter Mouton, Berlin, Germany, 2012.

Rubén San-Segundo, Juan Manuel Montero, Javier Macías-Guarasa, R Córdoba, Javier Ferreiros, and José Manuel Pardo. Proposing a speech to gesture translation architecture for Spanish deaf people. *Journal of Visual Languages & Computing*, 19(5):523–538, 2008.

Robert Smith and Brian Nolan. Emotional facial expressions in synthesised sign language avatars: A manual evaluation. *Universal Access in the Information Society*, pages 1–10, 2015. Retrieved from `http://dx.doi.org/10.1007/s10209-015-0410-7` (last accessed November 20, 2015).

J.C. Wells. SAMPA computer readable phonetic alphabet. In D. Gibbon, R. Moore, and R. Winski, editors, *Handbook of Standards and Resources for Spoken Language Systems*. De Gruyter Mouton, Berlin, Germany, 1997.

Ronnie B. Wilbur. Phonological and prosodic layering of nonmanuals in American Sign Language. In Karen Emmorey and Harlan Lane, editors, *The Signs of Language Revisited*, pages 215–244. Erlbaum, Mahwah, NJ, 2000.